Postel

Network Working Group
Request for Comments: 887

M. Accetta
Carnegie-Mellon University
December 1983

RESOURCE LOCATION PROTOCOL

This note describes a resource location protocol for use in the ARPA Internet. It is most useful on networks employing technologies which support some method of broadcast addressing, however it may also be used on other types of networks. For maximum benefit, all hosts which provide significant resources or services to other hosts on the Internet should implement this protocol. Hosts failing to implement the Resource Location Protocol risk being ignored by other hosts which are attempting to locate resources on the Internet. This RFC specifies a draft standard for the ARPA Internet community.

The Resource Location Protocol (RLP) utilizes the User Datagram Protocol (UDP) [1] which in turn calls on the Internet Protocol (IP) [3] to deliver its datagrams. See Appendix A and [6] for the appropriate port and protocol number assignments.

Unless otherwise indicated, all numeric quantities in this document are decimal numbers.

1. Introduction

From time to time, Internet hosts are faced with the problem of determining where on the Internet some particular network service or resource is being provided. For example, this situation will arise when a host needs to send a packet destined for some external network to a gateway on its directly connected network and does not know of any gateways. In another case, a host may need to translate a domain name to an Internet address and not know of any name servers which it can ask to perform the translation. In these situations a host may use the Resource Location Protocol to determine this information.

In almost all cases the resource location problem is simply a matter of finding the IP address of some one (usually any) host, either on the directly connected network or elsewhere on the Internet, which understands a given protocol. Most frequently, the querying host itself understands the protocol in question. Typically (as in the case of locating a name server), the querying host subsequently intends to employ that protocol to communicate with the located host once its address is known (e.g. to request name to address translations). Less frequently, the querying host itself does not necessarily understand the protocol in question. Instead (as in the case of locating a gateway), it is simply attempting to find some other host which does (e.g. to determine an appropriate place to forward a packet which cannot be delivered locally).

2. Resource Naming

Although the resource location problem can, in most cases, be reduced to the problem of finding a host which implements a given Internet based protocol, locating only a particular lowest level Internet protocol (i.e. one assigned a protocol number for transport using IP) is not completely sufficient. Many significant network services and resources are provided through higher level protocols which merely utilize the various lower level protocols for their own transport purposes (e.g. the FTP protocol [2] employs the TCP protocol [4] for its lower level transport). Conceptually, this protocol nesting may even be carried out to arbitrary levels.

Consequently, the Resource Location Protocol names a resource by the protocol number assigned to its lowest level Internet transport protocol and by a variable length protocol/resource specific identifier. For example, the UDP based Echo Protocol can be named by its assigned protocol number (17) and its assigned 16-bit "well-known" port number (7). Alternatively, the Internet Control Message Protocol [5] (lacking any higher level client protocols) would be named simply by its assigned protocol number (1) and an empty protocol specific identifier. On the other hand, some as yet undefined resource protocol (provided via say TCP), might be named by the assigned protocol number (6), its 16-bit "well-known" TCP port number, and then some additional fixed or variable length identifier specific to that TCP port.

In general, the components of the protocol/resource specific identifier are defined to be the "natural" quantities used to successively de-multiplex the protocol at each next highest protocol level. See section 5 for some sample assignments.

3. Protocol Summary

The Resource Location Protocol is a simple request/reply procedure. The querying host constructs a list of resources which it would like to locate and sends a request message on the network. A request message may be sent either to a particular IP address and host or, on networks which provide broadcast address capability, to the IP address which refers to all hosts on that network (see [7]). For example, a host attempting to locate a domain name server might construct a request containing the resource name [17, 53] (referring to the Domain Name Server protocol provided at "well-known" UDP port 53) and then broadcast that request on its local network.

Each receiving host examines the list of resources named in the request packet, determines which of the resources it provides, and returns a reply message to the querying host in confirmation. The receiving host determines whether or not it provides a resource by successive decomposition of the resource name until either the name is exhausted or it encounters a component which is not supported. In the previous

example, each host on the network receiving the broadcast request would examine the resource name by first consulting its tables to determine if it provided UDP service. If this was successful, it would then examine the UDP port component of the name and consult its UDP table to determine if it provided service on UDP port 53. At this point the name would be exhausted and if both checks were successful the host would return a reply message to the querying host indicating support for that resource.

3.1. Request Messages

RLP provides two basic types of request messages which may be transmitted by a querying host. The first type requires any host receiving the request message to return a reply message only if it provides at least one of the resources named in the request list. The second type requires any host receiving the message to always return a reply message even if it provides none of the resources named in the request list.

These two types of request messages are:

<Who-Provides?>
: This type requires any host receiving the message to return an appropriate reply message which names all of the resources from the request list which it provides. If the receiving host provides none of the named resources, no reply may be returned.

<Do-You-Provide?>
: This type is identical to the <Who-Provides?> message but with the extra requirement that a reply must always be returned. When a receiving host provides none of the requested resources, it simply returns an empty reply list. This empty reply list allows the querying host to immediately detect that the confirming host provides none of the named resources without having to timeout after repeatedly retransmitting the request.

The <Who-Provides?> request message is most typically used when broadcasting requests to an entire IP network. The <Do-You-Provide?> request message, on the other hand, is most typically used when confirming that a particular host does or does not provide one or more specific resources. It may not be broadcast (since such a request would flood the querying host with reply messages from all hosts on the network).

In addition to the two basic types of request messages, an additional two variant types of request messages may also be transmitted by a querying host. These message types provide a "third-party" resource location capability. They differ from the basic message types by

providing space for an additional qualifier with each listed resource to identify "third-party" hosts which the confirming host believes may provide the resource. As before, the first type requires any host receiving the request message to return a reply message only if it knows of some host which provides at least one of the resources named in the request list. The second type requires any host receiving the message to always return a reply message even if it knows of no hosts which provide any of the resources named in the request list.

These two remaining types of request messages are:

<Who-Anywhere-Provides?>
This message parallels the <Who-Provides?> message with the "third-party" variant described above. The confirming host is required to return at least its own IP address (if it provides the named resource) as well as the IP addresses of any other hosts it believes may provide the named resource. The confirming host though, may never return an IP address for a resource which is the same as an IP address listed with the resource name in the request message. In this case it must treat the resource as if it was unsupported at that IP address and omit it from any reply list.

<Does-Anyone-Provide?>
This message parallels the <Do-You-Provide?> message again with the "third-party" variant described above. As before, the confirming host is required to return its own IP address as well as the IP addresses of any other hosts it believes may provide the named resource and is prohibited from returning the same IP address in the reply resource specifier as was listed in the request resource specifier. As in the <Do-You-Provide?> case and for the same reason, this message also may not be broadcast.

These variant request messages permit "smart" hosts to supply resource location information for networks without broadcast capability (provided that all hosts on the network always "know" the address of one or more such "smart" hosts). They also permit resource location information for services which are not provided anywhere on a directly connected network to be provided by "smart" gateways which have perhaps queried other networks to which they are attached or have somehow otherwise acquired the information.

The restriction against returning the same IP address in a reply as was specified in the request provides a primitive mechanism for excluding certain known addresses from consideration in a reply (see section 5, example 3). It may also be used to override the receiving host's preference for its own IP address in "third-party" replies when this is required.

3.2. Reply Messages

Each of the types of request messages has an associated type of reply message. The basic reply message type is returned in response to both <Who-Provides?> and <Do-You-Provide?> request messages and supplies information about the resources provided by the confirming host. The other reply message type is the "third-party" variant returned in response to both <Who-Anywhere-Provides?> and <Does-Anyone-Provide?> request messages and supplies information about resources provided by hosts known to the confirming host.

These two types of reply messages are:

<I-Provide>
: This reply message contains a list of exactly those resources from the request list which the confirming host provides. These resources must occur in the reply list in precisely the same order as they were listed in the request message.

<They-Provide>
: This reply message similarly contains a list of exactly those resources from the request list (appropriately qualified with IP addresses) which the confirming host provides or believes another host provides. These resources again must occur in the reply list in precisely the same order as they were listed in the request message.

Neither type of reply message may be broadcast.

A querying host which receives a <They-Provide> reply message from a confirming host on behalf of a third host is not required to unquestioningly rely on the indirectly provided information. This information should usually be regarded simply as a hint. In most cases, the querying host should transmit a specific <Do-You-Provide?> request to the third host and confirm that the resource is indeed provided at that IP address before proceeding.

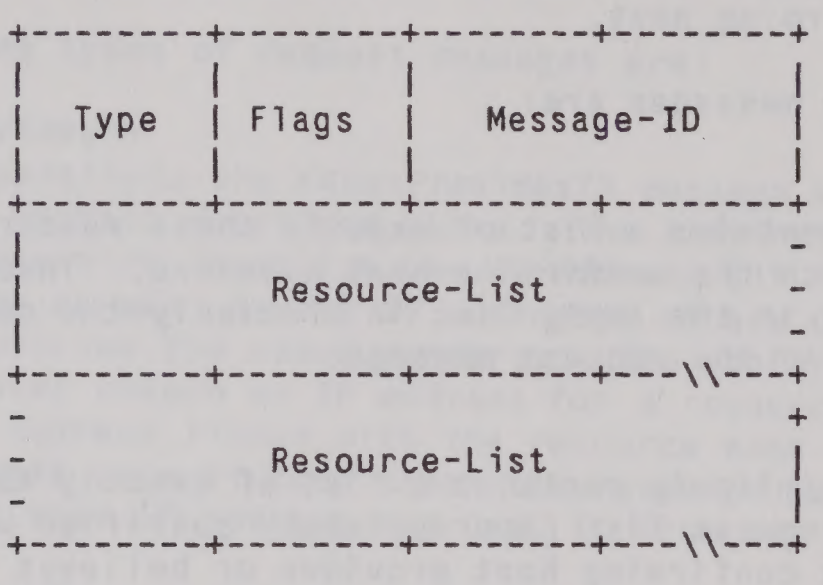
4. Message Formats

RLP messages are encapsulated in UDP packets to take advantage of the multiplexing capability provided by the UDP source and destination ports and the extra reliability provided by the UDP checksum. Request messages are sent from a convenient source port on the querying host to the assigned RLP destination port of a receiving host. Reply messages are returned from the assigned RLP source port of the confirming host to the appropriate destination port of the querying host as determined by the source port in the request message.

The format of the various RLP messages is described in the following diagrams. All numeric quantities which occupy more than one octet are

stored in the messages from the high order octet to the low order octet as per the usual Internet protocol standard. All packet diagrams indicate the octets of the message from left to right and then top to bottom as they occur in the data portion of the encapsulating UDP packet.

Each RLP packet has the general format



where

<Type>
is a single octet which identifies the message type. The currently defined types are:

0 <Who-Provides?>
1 <Do-You-Provide?>
2 <Who-Anywhere-Provides?>
3 <Does-Anyone-Provide?>
4 <I-Provide>
5 <They-Provide>
6-255 Reserved.

<Flags>
is a single octet specifying possible modifications to the standard interpretation of <Type>. Bits in this field are numbered from left to right (from most significant to least significant) beginning with bit 1. The currently defined flag bits are:

Bit 1 <Local-Only>. Requires that any reply message generated in response to a request message with this flag bit set may only name IP addresses which are on the same IP network as the sender of the request message. This flag also requires that multi-homed hosts answering broadcast <Who-Provides?> requests use the appropriate local network IP source address in the returned reply. This bit must be zero in reply messages.

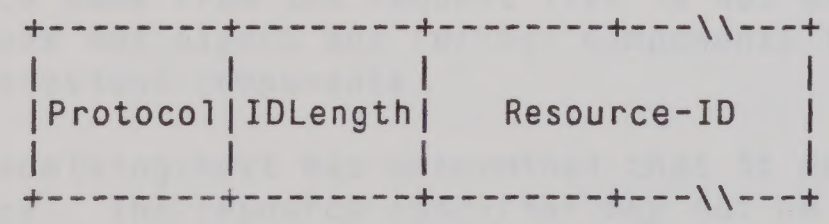
Bits 2-8 Reserved. Must be zero.

<Message-ID>
is a two octet (16-bit) value which identifies the request message. It is used simply to aid in matching requests with replies. The sending host should initialize this field to some convenient value when constructing a request message. The receiving host must return this same value in the <Message-ID> field of any reply message generated in response to that request.

<Resource-List>
is the list of resources being queried or for which location information is being supplied. This list is a sequence of octets beginning at the octet following the <Message-ID> and extending through the end of the UDP packet. The format of this field is explained more fully in the following section. The size of this list is implicitly specified by the length of the encapsulating UDP datagram.

4.1. Resource Lists

A <Resource-List> consists of zero or more resource specifiers. Each resource specifier is simply a sequence of octets. All resource specifiers have a common resource name initial format



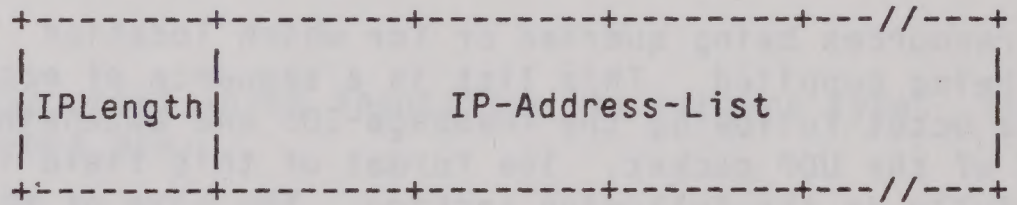
where

<Protocol>
is the protocol number assigned to the lowest level Internet protocol utilized for accessing the resource.

<IDLength>
is the length of the resource identifier associated with this <Protocol>. This length may be a fixed or variable value depending on the particular resource. It is included so that specifiers which refer to resources which a host may not provide can be skipped over without needing to know the specific structure of the particular resource identifier. If the <Protocol> has no associated natural identifier, this length is 0.

<Resource-ID>
is the qualifying identifier used to further refine the resource being queried. If the <IDLength> field was 0, then this field is empty and occupies no space in the resource specifier.

In addition, resource specifiers in all <Who-Anywhere-Provides?>, <Does-Anyone-Provide?> and <They-Provide> messages also contain an additional qualifier following the <Protocol-ID>. This qualifier has the format



where

<IPLength>
is the number of IP addresses containing in the following <IP-Address-List> (the <IP-Address-List> field thus occupies the last 4*<IPLength> octets in its resource specifier). In request messages, this is the maximum number of qualifying addresses which may be included in the corresponding reply resource specifier. Although not particularly useful, it may be 0 and in that case provides no space for qualifying the resource name with IP addresses in the returned specifier. In reply messages, this is the number of qualifying addresses known to provide the resource. It may not exceed the number specified in the corresponding request specifier. This field may not be 0 in a reply message unless it was supplied as 0 in the request message and the confirming host would have returned one or more IP addresses had any space been provided.

<IP-Address-List>
is a list of four-octet IP addresses used to qualify the resource specifier with respect to those particular addresses. In reply messages, these are the IP addresses of the confirming host (when appropriate) and the addresses of any other hosts known to provide that resource (subject to the list length limitations). In request messages, these are the IP addresses of hosts for which resource information may not be returned. In such messages, these addresses should normally be initialized to some "harmless" value (such as the address of the querying host) unless it is intended to specifically exclude the supplied addresses from consideration in any reply messages.

The receiving host determines if it provides any of the resources named in the request list by successively decomposing each resource name. The first level of decomposition is the Internet protocol number which can presumably be looked up in a table to determine if that protocol is supported on the host. Subsequent decompositions are based on previous components until one of three events occur:

1. the current component identifies some aspect of the previous components which the host does not support,

2. the resource name from the request list is exhausted, or

3. the resource name from the request list is not exhausted but the host does not expect any further components in the name given the previous components

In case 1, the receiving host has determined that it does not provide the named resource. The resource specifier may not be included in any reply message returned.

In case 2, the receiving host has determined that it does indeed provide the named resource (note: this may occur even if the receiving host

would have expected the resource name to contain more components than were actually present). The resource specifier must be included (modulo IP address prohibitions) in any reply message returned.

In case 3, the receiving host has determined that it does not completely provide the named resource since name components remain which it does not understand (this might occur with specializations of or extensions to a known protocol which are not universally recognized). The resource specifier may not be included in any reply message returned.

5. Sample Usage

The following scenarios illustrate some typical uses of RLP. In all cases the indicated messages are encapsulated in a UDP datagram with the appropriate source and destination port numbers, message length, and checksum. This datagram is further encapsulated in an IP datagram with the appropriate source address of the sending host and destination address (either broadcast or individual) for the receiving host.

All numeric protocol examples are as specified in the appropriate protocol description documents listed in the references.

1. Suppose a freshly rebooted host H wishes to find some gateway on its directly connected network to which it can send its first external packet. It then broadcasts the request

   <Who-Provides?> <Flags>=<Local-Only> <Message-ID>=12345
   <Resource-List>={[GGP], [EGP]}

   encoded as the 8 octet message

   | 0 | 128 | 12345 | 3 | 0 | 8 | 0 |

   on its local network.

   - Gateway G1 (which understands EGP) receives the request and returns the reply

     <I-Provide> <Flags>=none <Message-ID>=12345
     <Resource-List>={[EGP]}

     encoded as the 6 octet message

     | 4 | 0 | 12345 | 8 | 0 |

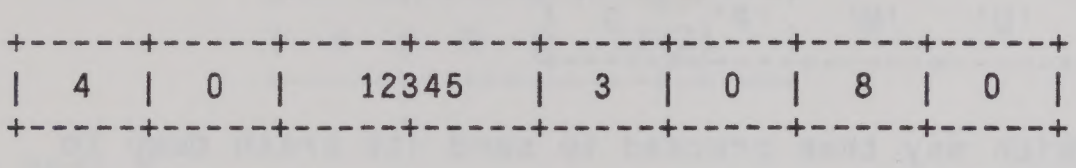
     to host H which then remembers that gateway G1 may be used

to route traffic to the rest of the Internet.

- At the same time, gateway G2 (which understands both GGP and EGP) might also receive the request and return the reply

<I-Provide> <Flags>=none <Message-ID>=12345
<Resource-List>={[GGP], [EGP]}

encoded as the 8 octet message

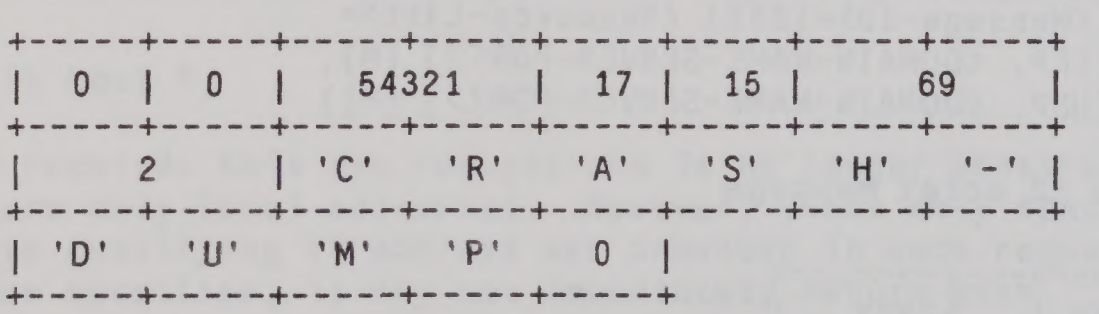


to host H which might then also add gateway G2 to its list if it chooses.

2. Assume instead that host H is a stand-alone system which has just encountered some fatal software error and wishes to locate a crash dump server to save its state before reloading. Suppose that the crash dump protocol on the host's local network is implemented using the Trivial File Transfer Protocol (TFTP) [8]. Furthermore, suppose that the special file name "CRASH-DUMP" is used to indicate crash dump processing (e.g. the server might locally generate a unique file name to hold each dump that it receives from a host). Then host H might broadcast the request

<Who-Provides?> <Flags>=none <Message-ID>=54321
<Resource-List>={[UDP, TFTP, WRQ, "CRASH-DUMP"]}

encoded as the 21 octet message

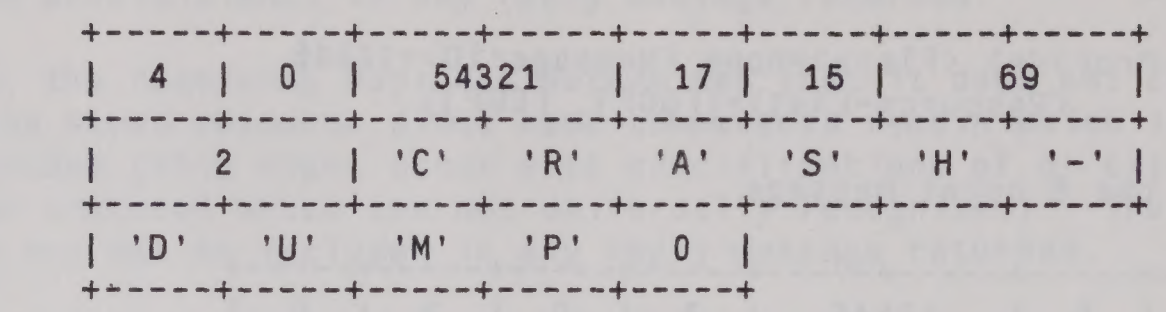


to its local network (note that the file name component is explicitly terminated by a null so as not to exclude future further specialization of the crash dump protocol).

- Host C (which supports this specialization of the TFTP protocol) receives the request and returns the reply

<I-Provide> <Flags>=none <Message-ID>=54321
<Resource-List>={[UDP, TFTP, WRQ, "CRASH-DUMP"]}

encoded as the 21 octet message



to host H which may then proceed to send its crash dump to host C and reload.

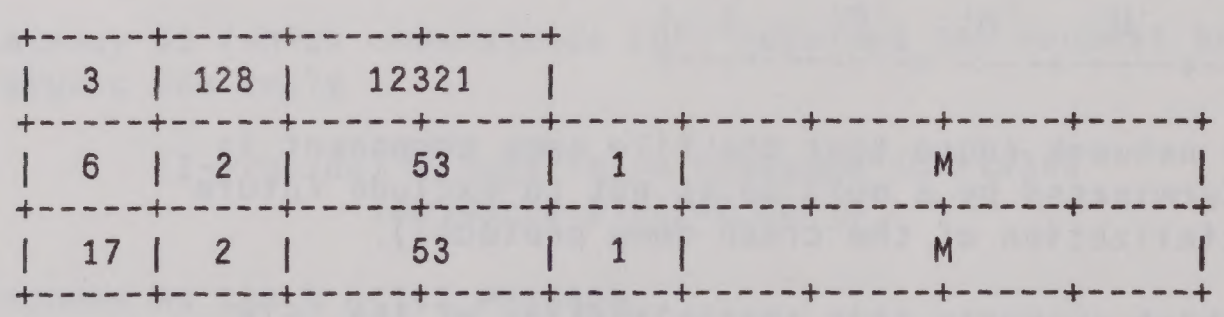
- Host D (which provides TFTP service but not the crash dump specialization), however, might receive the request and determine that it provides no support for the resource (since the resource name contains components following the UDP port number which it does not understand). It would therefore return no reply to host H.

3. Finally, suppose host M wishes to locate some domain name translation server (either UDP or TCP based) anywhere on the Internet. Furthermore, suppose that host M is on a IP network which does not provide broadcast address capabilities and that host R is a "known" resource location server for that network.

   First, since host M prefers to find a domain name server on its own locally connected network if possible, it sends the request

```
<Does-Anyone-Provide?> <Flags>=<Local-Only>
    <Message-ID>=12321 <Resource-List>=
  {[TCP, <DOMAIN-NAME-SERVER-PORT>] {M},
   [UDP, <DOMAIN-NAME-SERVER-PORT>] {M}}
```

   encoded as the 22 octet message



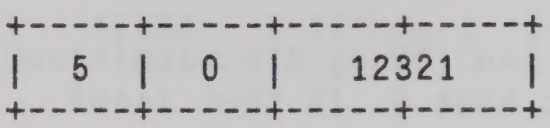
   to host R.

   Host R receives the request and consults its tables for any hosts known to support either variety of domain name service. It finds entries indicating that both hosts S and T provide UDP

based domain name service but that neither host is on the same IP network as host H. It must then send the negative confirmation reply

<They-Provide> <Flags>=none <Message-ID>=12321
<Resource-List>={}

encoded as the 4 octet message

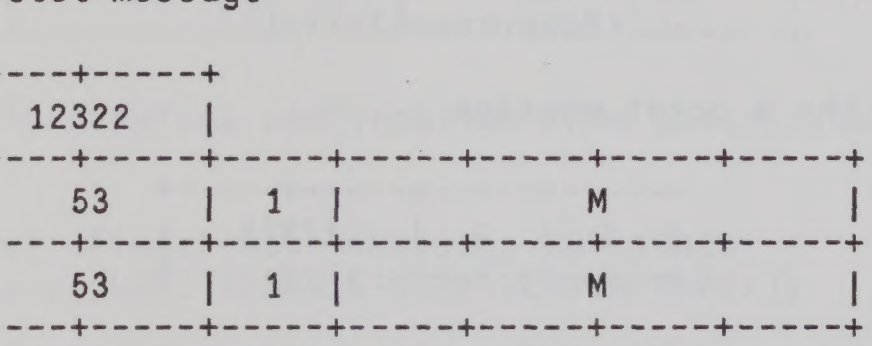


back to host M.

Host M, receiving this reply, might now abandon any hope of finding a server on its own network, reformat its request to permit any host address, and resend

<Does-Anyone-Provide?> <Flags>=none <Message-ID>=12322
<Resource-List>=
{[TCP, <DOMAIN-NAME-SERVER-PORT>] {M},
[UDP, <DOMAIN-NAME-SERVER-PORT>] {M}}

encoded as the 22 octet message

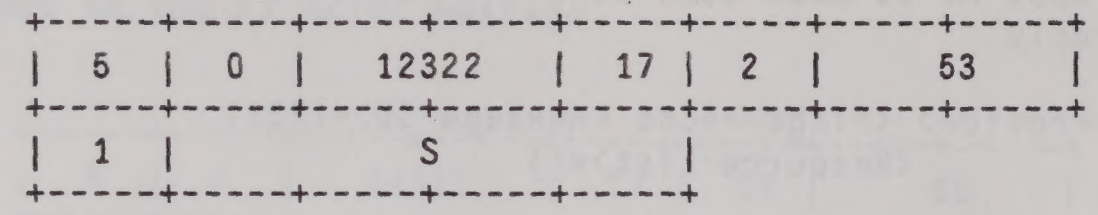


again to host R.

Host R receives this new request and is no longer constrained to return only local addresses. However, since only space for a single qualifying IP address was provided in each request resource specifier, it may not immediately return both addresses. Instead, it is forced to return only the first address and replies

<They-Provide> <Flags>=none <Message-ID>=12322
<Resource-List>={[UDP, <DOMAIN-NAME-SERVER-PORT>] {S}}

encoded as the 13 octet message

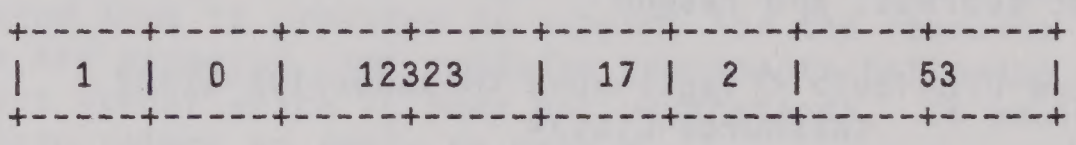


to Host M.

Host M receives the reply and (being the suspicious sort) decides to confirm it with host S. It then sends

<Do-You-Provide?> <Flags>=none <Message-ID>=12323
<Resource-List>={[UDP, <DOMAIN-NAME-SERVER-PORT>]}

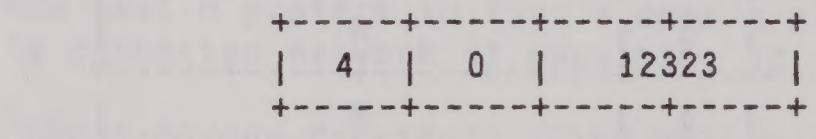
encoded as the 8 octet message



to host S and receives back from host S the reply

<I-Provide> <Flags>=none <Message-ID>=12323
<Resource-List>={}

encoded as the 4 octet message

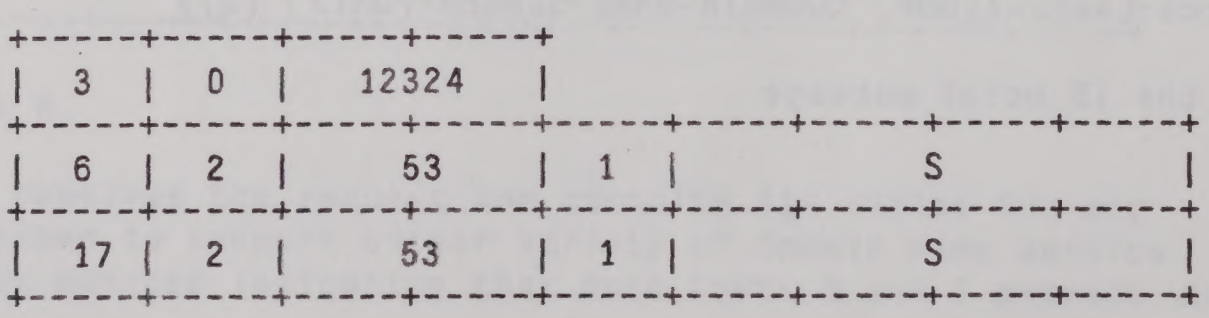


denying any support for UDP based domain name service.

In desperation host M again queries host R, this time excluding host S from consideration, and sends the request

<Does-Anyone-Provide?> <Flags>=none <Message-ID>=12324
<Resource-List>=
{[TCP, <DOMAIN-NAME-SERVER-PORT>] {S},
[UDP, <DOMAIN-NAME-SERVER-PORT>] {S}}

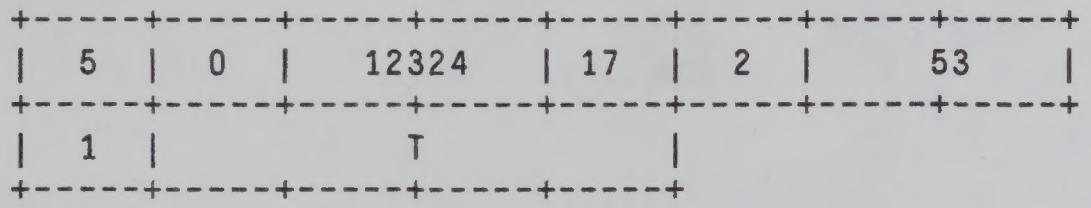
encoded as the 22 octet message



and this time receives the reply

<They-Provide> <Flags>=none <Message-ID>=12324
<Resource-List>={[UDP, <DOMAIN-NAME-SERVER-PORT>] {T}}

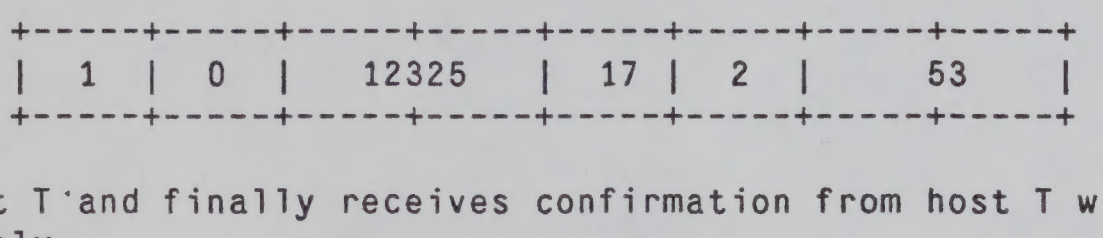
encoded as the 13 octet message



from host R which of course host M again insists on confirming by sending the request

<Do-You-Provide?> <Flags>=none <Message-ID>=12325
<Resource-List>=
{[UDP, <DOMAIN-NAME-SERVER-PORT>]}

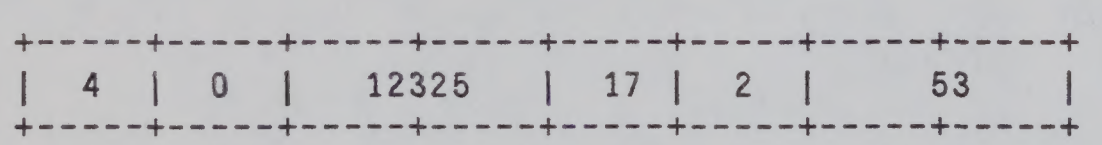
encoded as the 8 octet message



to host T and finally receives confirmation from host T with the reply

<I-Provide> <Flags>=none <Message-ID>=12325
<Resource-List>={[UDP, <DOMAIN-NAME-SERVER-PORT>]}

encoded as the 8 octet message



that it indeed provides domain name translation service at UDP port 53.

A. Assigned Numbers

The "well-known" UDP port number for the Resource Location Protocol is 39 (47 octal).

REFERENCES

[1] Postel, J.
User Datagram Protocol.
RFC 768, USC/Information Sciences Institute, August, 1980.

[2] Postel, J.
File Transfer Protocol.
RFC 765, USC/Information Sciences Institute, June, 1980.

[3] Postel, J.
Internet Protocol - DARPA Internet Program Protocol Specification.
RFC 791, USC/Information Sciences Institute, September, 1981.

[4] Postel, J.
Transmission Control Protocol- DARPA Internet Program Protocol Specification.
RFC 793, USC/Information Sciences Institute, September, 1981.

[5] Postel, J.
Internet Control Message Protocol - DARPA Internet Program Protocol Specification.
RFC 792, USC/Information Sciences Institute, September, 1981.

[6] Reynolds, J., and J. Postel.
Assigned Numbers.
RFC 870, USC/Information Sciences Institute, October, 1983.

[7] Gurwitz, R., and R. Hinden.
IP - Local Area Network Addressing Issues.
IEN 212, Bolt Beranek and Newman, September, 1982.

[8] Sollins, K.
The TFTP Protocol (revision 2).
RFC 783, MIT/Laboratory for Computer Science, June, 1981.